EBIO 3080 Week 1 R Tutorial

Silas Tittes

23 August, 2016

Goals

- Basics

Goals

- Basics
- Navigating your computer like a pro (sort of) – file paths with getwd(), setwd(), and list.files()

Goals

- Basics
- Navigating your computer like a pro (sort of) – file paths with getwd(), setwd(), and list.files()
- Read data from a text file or csv into R – read.csv() or read.table()

Goals

- Basics
- Navigating your computer like a pro (sort of) – file paths with getwd(), setwd(), and list.files()
- Read data from a text file or csv into R – read.csv() or read.table()
- Working with data frames – the $ operator

Goals

- Basics
- Navigating your computer like a pro (sort of) – file paths with getwd(), setwd(), and list.files()
- Read data from a text file or csv into R – read.csv() or read.table()
- Working with data frames – the $ operator
- Basic math and summary statistics on vectors

Goals

- Basics
- Navigating your computer like a pro (sort of) – file paths with getwd(), setwd(), and list.files()
- Read data from a text file or csv into R – read.csv() or read.table()
- Working with data frames – the $ operator
- Basic math and summary statistics on vectors
- Plotting data – plot() and hist()

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.
- To run a command, use the "Control" + "Enter" keys, line does not need to be highlighted, but multiple lines can be highlighted if you want to run multiple lines.

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.
- To run a command, use the "Control" + "Enter" keys, line does not need to be highlighted, but multiple lines can be highlighted if you want to run multiple lines.
- Assignment operator to create a new variable/object "<-" or "=".

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.
- To run a command, use the "Control" + "Enter" keys, line does not need to be highlighted, but multiple lines can be highlighted if you want to run multiple lines.
- Assignment operator to create a new variable/object "<-" or "=".
- Hotkey for making "<-" is "Alt" + "-" keys.

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.
- To run a command, use the "Control" + "Enter" keys, line does not need to be highlighted, but multiple lines can be highlighted if you want to run multiple lines.
- Assignment operator to create a new variable/object "<-" or "=".
- Hotkey for making "<-" is "Alt" + "-" keys.
- Use "Control" + "a" to highlight all code, then "Control" + "Enter" to run all code.

Basics

- Write all commands in a scipt and save with .R extension. Sparingly use the console directly.
- To run a command, use the "Control" + "Enter" keys, line does not need to be highlighted, but multiple lines can be highlighted if you want to run multiple lines.
- Assignment operator to create a new variable/object "<-" or "=".
- Hotkey for making "<-" is "Alt" + "-" keys.
- Use "Control" + "a" to highlight all code, then "Control" + "Enter" to run all code.
- Use the "tab" key and `R` will often complete or predict the remaining text. *Very useful*!

Navigating your computer like a pro

To be effecient in R we need to know where we are and where we want to go within our computer.

```
#try using the tab button for
#suggestions on how to finish your path

#get working directory
getwd()
```

```
[1] "/home/silastittes/Documents/TAing/Fall16_Evolution/wee
```

```
#set working directory
setwd("/home/silastittes/Desktop/")
getwd()
```

```
[1] "/home/silastittes/Desktop"
```

```
setwd("~/Documents/TAing/Fall16_Evolution/week1/")
```

Navigating your computer like a pro

Make sure your files are where you think they are!

```
list.files()
```

```
[1] "Height_Fall2014_v2.csv" "horned_lizards.csv"
[3] "Week1_RTutorial.pdf"    "Week1_RTutorial.Rmd"
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.
- Use read.table() to read in a txt file.

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.
- Use read.table() to read in a txt file.
- Once you know where yourself and the data files you need are, read data into R.

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.
- Use read.table() to read in a txt file.
- Once you know where yourself and the data files you need are, read data into R.
- Expect data to almost always be in csv format for this class.

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.
- Use read.table() to read in a txt file.
- Once you know where yourself and the data files you need are, read data into R.
- Expect data to almost always be in csv format for this class.
- Always save data to an object, with a meaningful name.

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Read data from a text file or csv into R

- Use read.csv() to read in a csv file.
- Use read.table() to read in a txt file.
- Once you know where yourself and the data files you need are, read data into R.
- Expect data to almost always be in csv format for this class.
- Always save data to an object, with a meaningful name.
- Let R know the first row of the data set are names for the columns using "header = TRUE"

```
lizard <- read.csv("horned_lizards.csv", header = TRUE)
```

Working with data frames

Take a quick look at the data with head(), which prints the first six rows be default.

```
head(lizard)
```

```
  squamosalHornLength Survival
1                25.2   living
2                26.9   living
3                26.6   living
4                25.6   living
5                25.7   living
6                25.9   living
```

Working with data frames

Work with individual columns of the data frame with the $ operator.

```
head(lizard$Survival)
```

```
[1] living living living living living living
Levels: killed living
```

```
head(lizard$squamosalHornLength)
```

```
[1] 25.2 26.9 26.6 25.6 25.7 25.9
```

Working with data frames

Get rows or columns of the data frame with indexing brackets

```
#first row all columns
lizard[1,]
```

```
  squamosalHornLength Survival
1                25.2   living
```

```
#second row first column
lizard[2,1]
```

```
[1] 26.9
```

```
#last row second column
lizard[nrow(lizard),2]
```

```
[1] killed
Levels: killed living
```

Basic math and summary statistics on vectors

- R likes vectors – 1 x n dimnsional sets of values all of the same type.
- Use the c() function to construct a new vectors.

```
#numeric vector
n <- c(1, 2, 3)
n
```

```
[1] 1 2 3
```

```
is.vector(n) #if a vector, will print TRUE
```

```
[1] TRUE
```

```
#character vector
ch <- c("a", "b", "c")
ch
```

```
[1] "a" "b" "c"
```

Basic math and summary statistics on vectors

Math operators are pretty straight forward.

```
n + n
```

```
[1] 2 4 6
```

```
n * n
```

```
[1] 1 4 9
```

```
n^2
```

```
[1] 1 4 9
```

```
sqrt(n)
```

```
[1] 1.000000 1.414214 1.732051
```

```
log(n)
```

```
[1] 0.0000000 0.6931472 1.0986123
```

Basic math and summary statistics on vectors

R thinks of columns of a data frame as a vector, so rules for vectors usually apply to columns of a data frame.

```
#don't use head!
head(lizard$squamosalHornLength)
```

```
[1] 25.2 26.9 26.6 25.6 25.7 25.9
```

```
head(lizard$squamosalHornLength * 2)
```

```
[1] 50.4 53.8 53.2 51.2 51.4 51.8
```

Basic math and summary statistics on vectors

R has an immense statistical library

```
mean(lizard$squamosalHornLength) #DOH!
```

```
[1] NA
```

```
mean(lizard$squamosalHornLength, na.rm = TRUE)
```

```
[1] 23.90707
```

```
median(lizard$squamosalHornLength, na.rm = TRUE)
```

```
[1] 24.15
```

```
var(lizard$squamosalHornLength, na.rm = TRUE)
```

```
[1] 7.672136
```

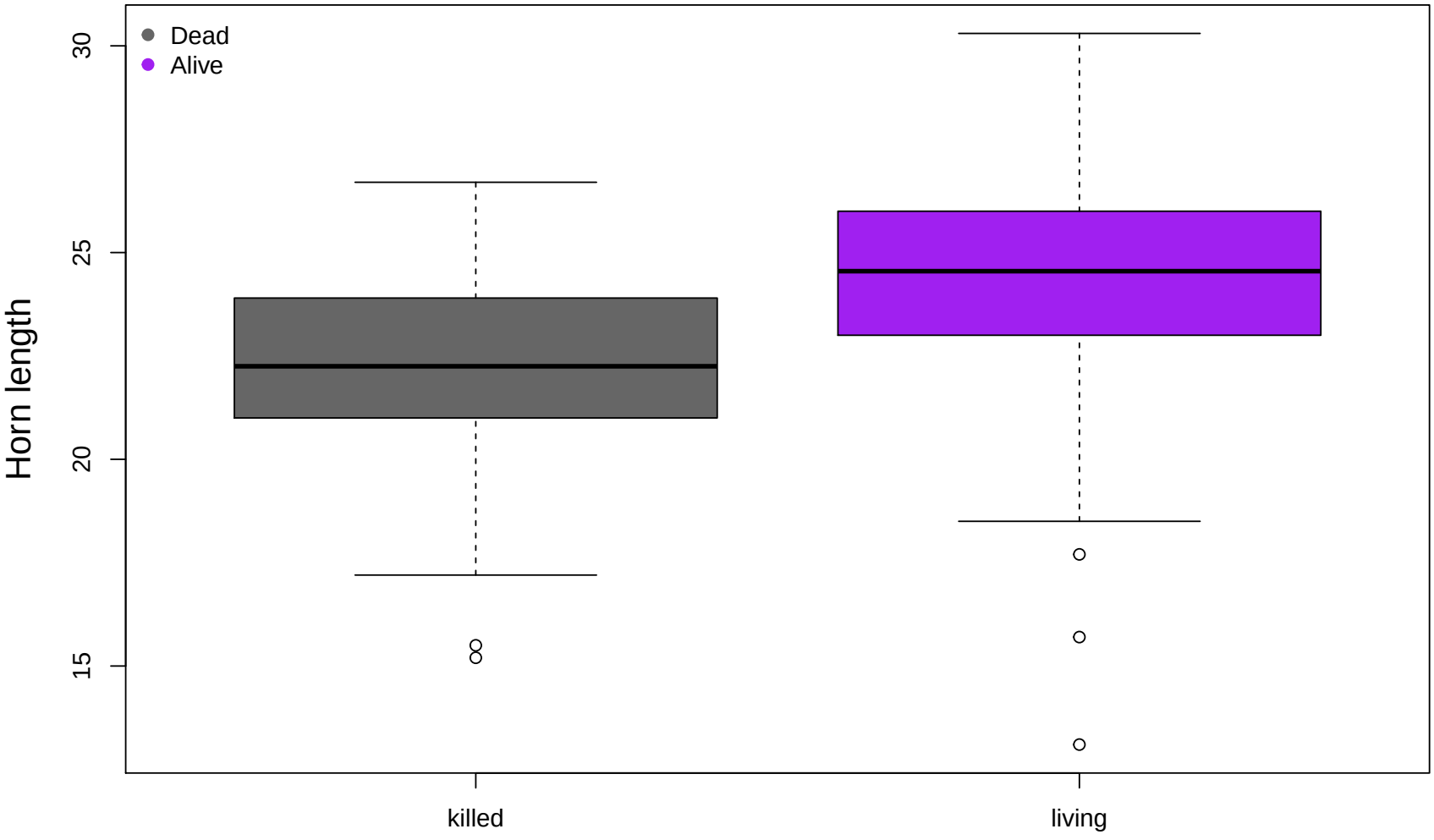
```
sd(lizard$squamosalHornLength, na.rm = TRUE)
```

Plotting data

R also has some incredible graphics utilities

```
boxplot(lizard$squamosalHornLength ~ lizard$Survival,
        ylab = "Horn length",
        col=c("grey40", "purple"),
        cex.lab=1.4)

legend("topleft", c("Dead", "Alive"),
       pch=19, col=c("grey40", "purple"), bty="n")
```

Plotting data

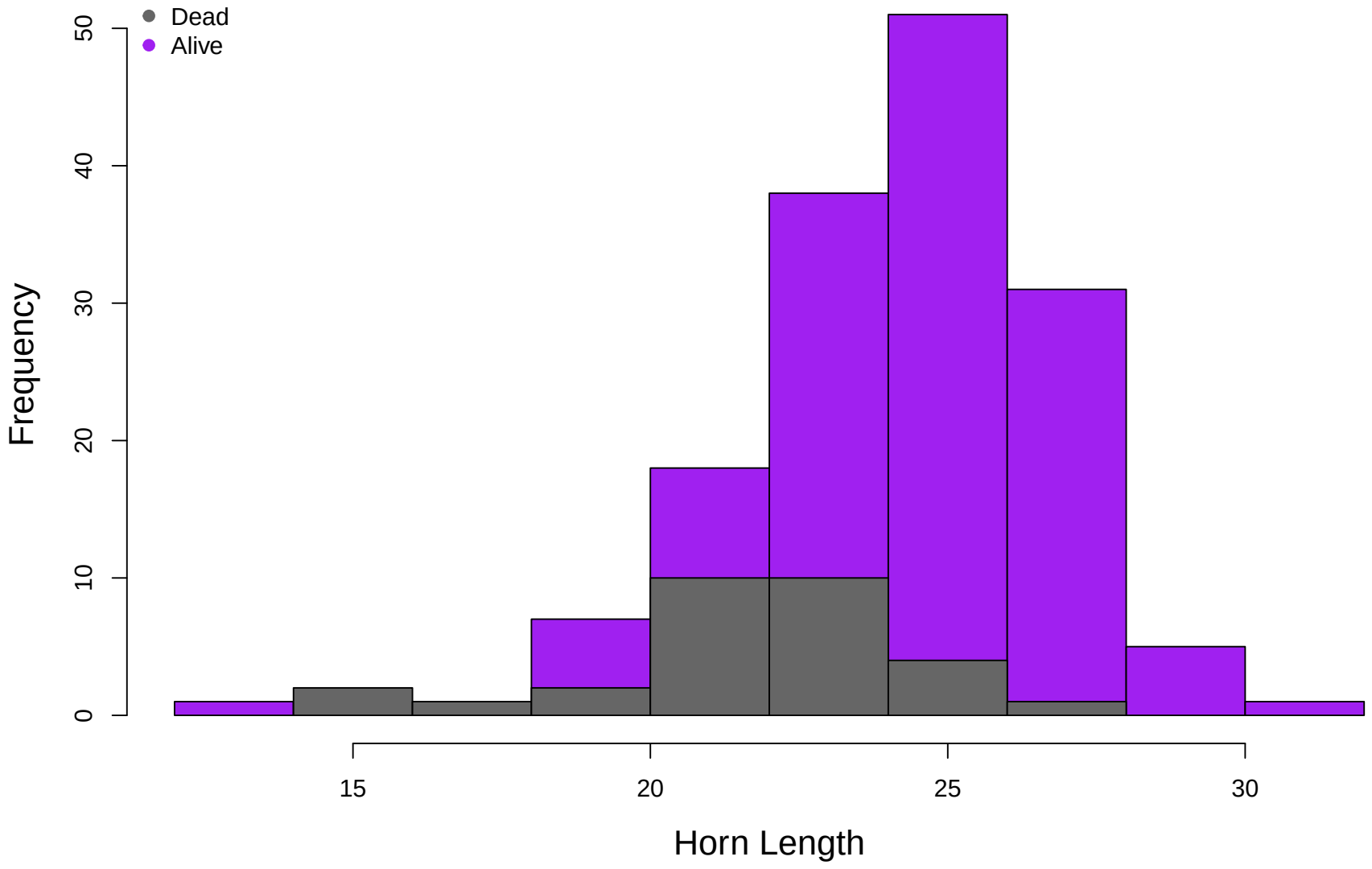


Plotting data

R also has some incredible graphics utilities

```
dead <- lizard[lizard$Survival == "killed", ]
alive <- lizard[lizard$Survival != "killed", ]

rng <- range( c(dead$squamosalHornLength, alive$squamosalHo

hist(alive$squamosalHornLength,
     xlab = "Horn Length",
     cex.lab=1.4,
     cex=0.5, main = "", col = "purple")

hist(dead$squamosalHornLength, xlim=rng,
     xlab = "Horn Length",
     cex.lab=1.4,
     cex=0.5, main = "", col = "grey40", add=T)


legend("topleft", c("Dead", "Alive"),
```

Plotting data

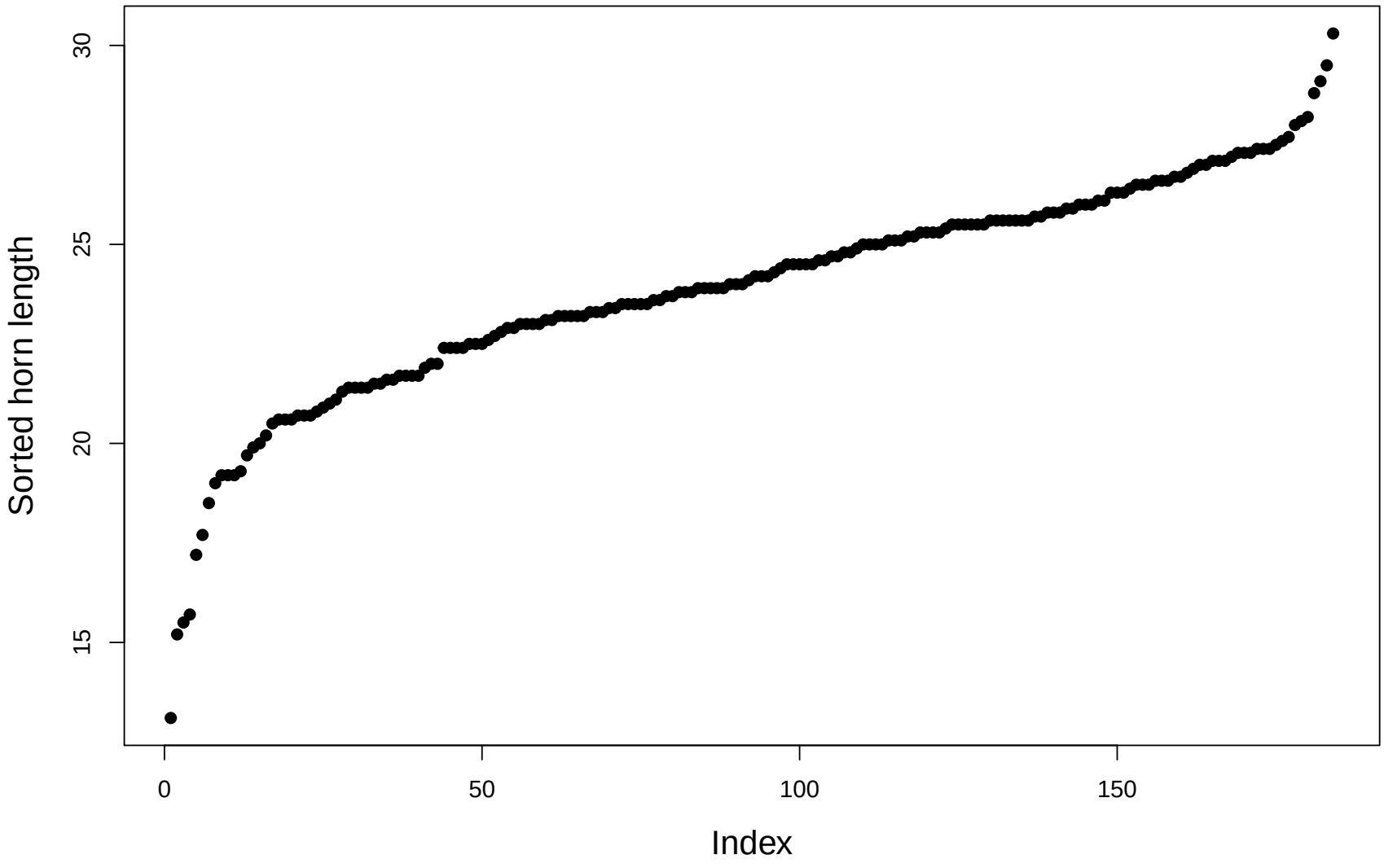


Plotting data

R also has some incredible graphics utilities

```
plot(sort(lizard$squamosalHornLength),
     ylab="Sorted horn length",
     pch=19, cex.lab=1.4)
```

Plotting data



Plotting data

R also has some incredible graphics utilities

```
surv <- as.numeric(lizard$Survival)-1
colz <- ifelse(surv == 1, "grey40", "purple")
plot(lizard$squamosalHornLength, jitter(surv, 0.1),
     xlab="Sorted horn length", ylab = "Survival",
     pch=19, col = colz, cex.lab=1.4)


legend("topleft", c("Dead", "Alive"),
       pch=19, col=c("grey40", "purple"), bty="n")
```

Plotting data